AES

Cifras de Bloco

Prof. Dr. Iaçanã Ianiski Weber

Confiabilidade e Segurança de Software

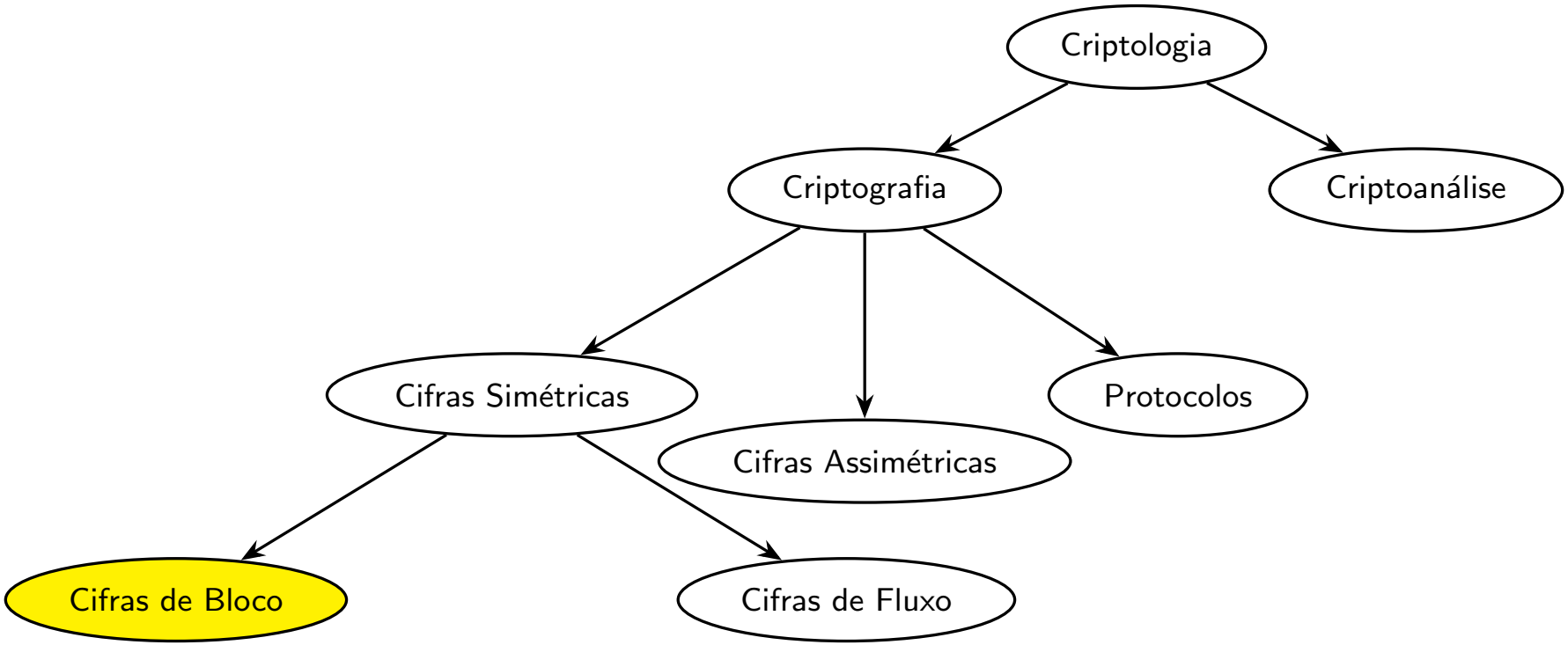
98G08-4

Agradecimentos especiais ao Prof. Avelino Zorzo e aos Autores Christof Paar e Jan Pelzl pelo material.

Índice

Índice

AES no campo da Criptologia



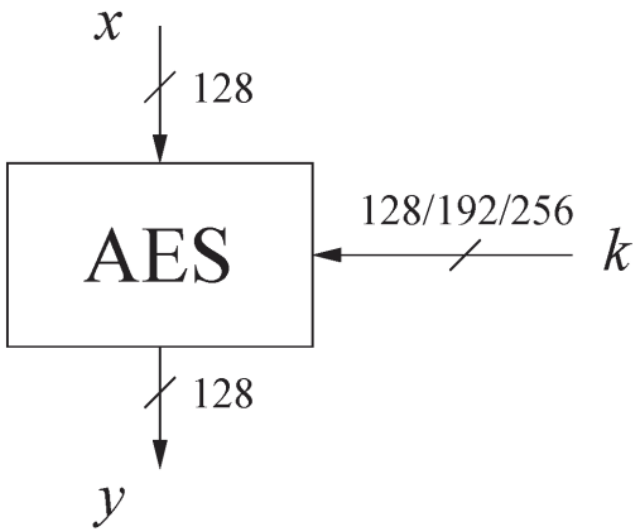
Fatos Básicos sobre o AES

- O AES é o cifrador simétrico mais utilizado atualmente.
- O algoritmo AES foi escolhido pelo *National Institute of Standards and Technology* (NIST) dos EUA em um processo de seleção que durou vários anos.
- Os requisitos para todas as submissões candidatas ao AES eram:
  - Cifra de bloco com **tamanho de bloco de 128 bits**.
  - **Três tamanhos de chave suportados**: 128, 192 e 256 bits.
  - Segurança relativa em comparação com outros algoritmos submetidos.
  - **Eficiência** em software e hardware.

Cronologia da Seleção do AES

- A necessidade de uma nova cifra de bloco foi anunciada pelo NIST em janeiro de 1997.
- 15 algoritmos candidatos foram aceitos em agosto de 1998.
- 5 finalistas foram anunciados em agosto de 1999:
  - *Mars* – IBM Corporation
  - *RC6* – RSA Laboratories
  - *Rijndael* – J. Daemen & V. Rijmen
  - *Serpent* – Eli Biham et al.
  - *Twofish* – B. Schneier et al.
- Em outubro de 2000, o *Rijndael* foi escolhido como o AES.
- O AES foi formalmente aprovado como padrão federal dos EUA em novembro de 2001.

AES: Visão Geral

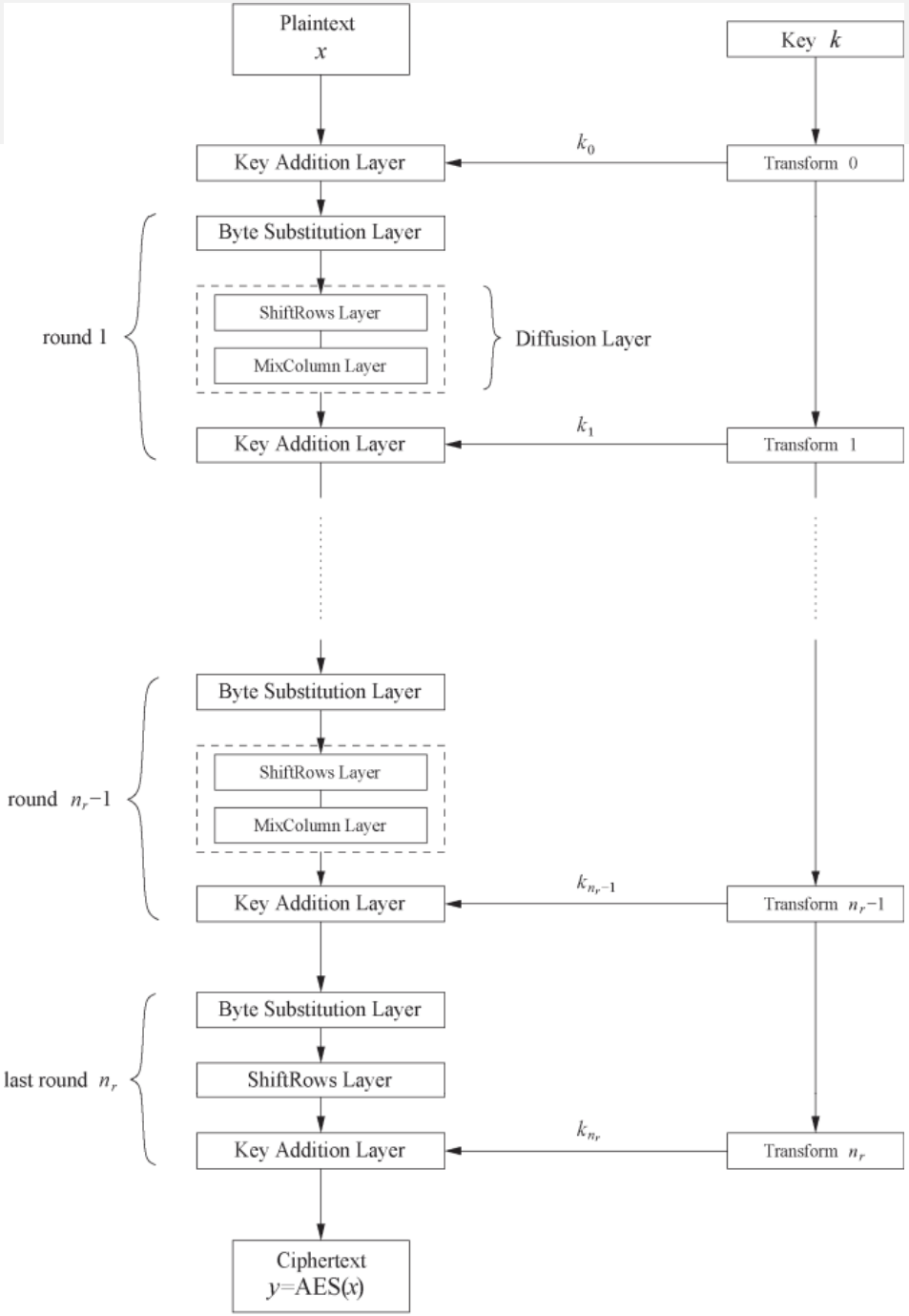


O número de rodadas depende do tamanho da chave escolhida:

Tamanho da chave (bits)	Número de rodadas
128	10
192	12
256	14

AES: Visão Geral

- Cifra iterada com 10/12/14 rodadas
- Cada rodada consiste em "Camadas"



Índice

Estrutura Interna do AES

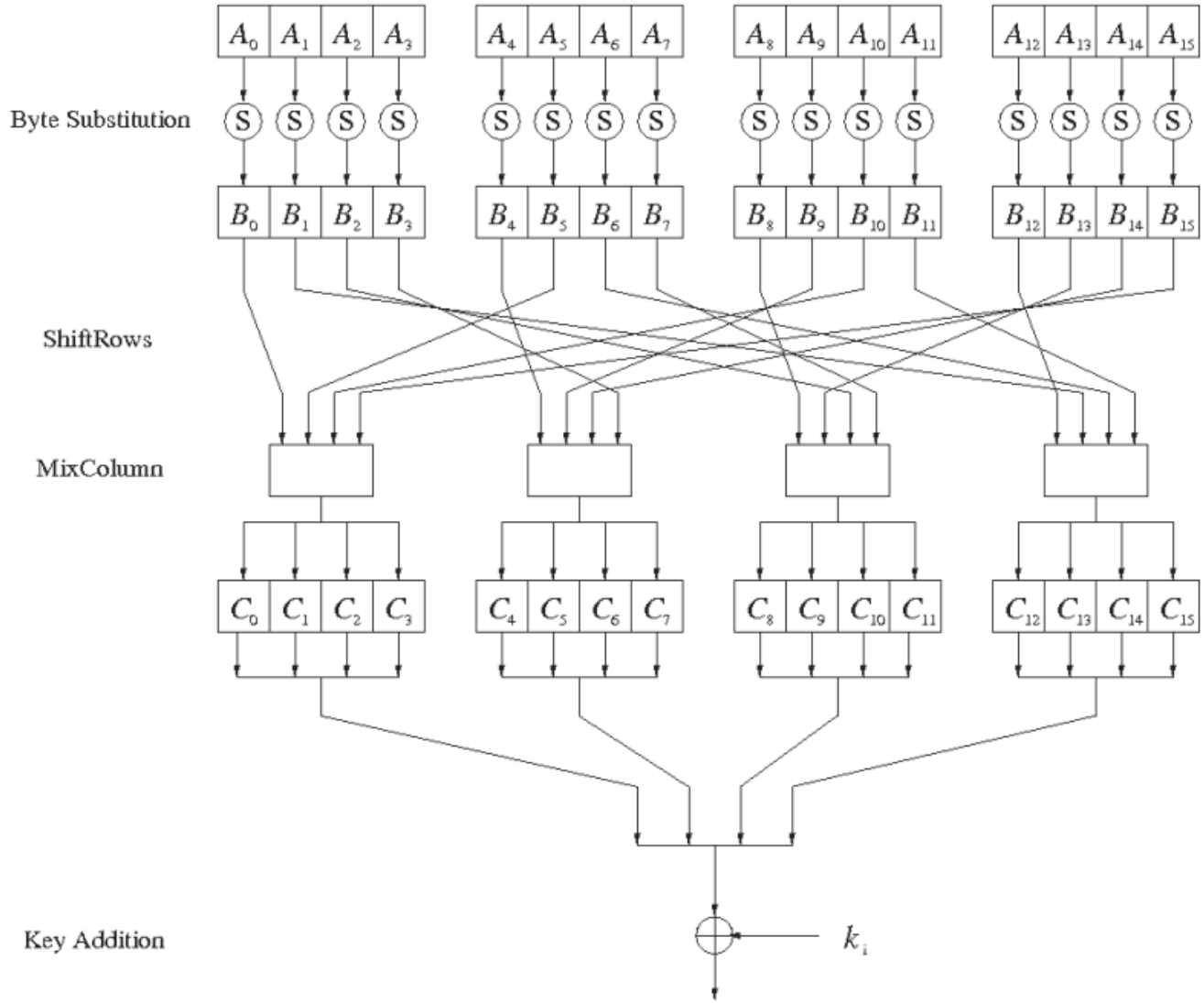
- AES é um cifra orientada a bytes
- O estado A (isto é, o caminho de dados de 128 bits) pode ser organizado em uma matriz 4x4:

$$\begin{bmatrix} A_0 & A_4 & A_8 & A_{12} \\ A_1 & A_5 & A_9 & A_{13} \\ A_2 & A_6 & A_{10} & A_{14} \\ A_3 & A_7 & A_{11} & A_{15} \end{bmatrix}$$

com $A_0, \ldots, A_{15}$ denotando os 16 bytes de entrada do AES.

Estrutura Interna do AES

- Função de rodada para as rodadas $1, 2, \ldots, n_r - 1$:



- Nota: Na última rodada, a transformação **MixColumn** é omitida

Camada de Substituição de Byte

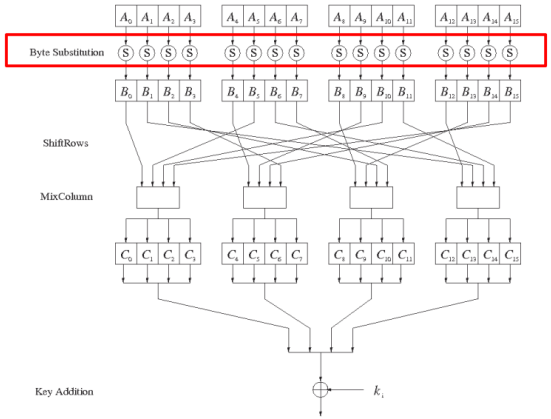
- A camada Byte Substitution consiste em 16 **S-Boxes** com as seguintes propriedades:
  - As S-Boxes são
    - **idênticas**
    - o único elemento *não linear* do AES, ou seja,

      $$\text{ByteSub}(A_i)+\text{ByteSub}(A_j) \neq \text{ByteSub}(A_i+A_j), \quad i,j = 0,\ldots,15$$

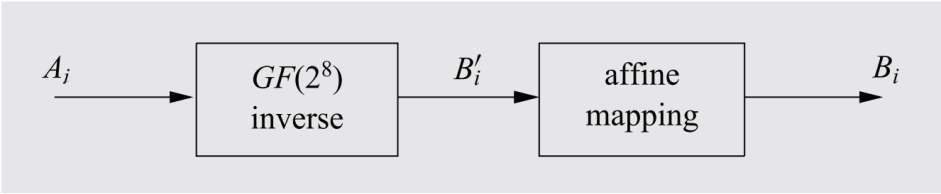
    - **bijetivas**, isto é, existe uma correspondência um-para-um entre bytes de entrada e saída

      $\Rightarrow$ A S-Box pode ser invertida de forma única

- Em implementações, a S-Box é geralmente realizada como uma *lookup table*



AES GF(2^8) Inverso Multiplicativo



X \ Y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	8D	F6	CB	52	7B	D1	E8	4F	29	C0	B0	E1	E5	C7
1	74	B4	AA	4B	99	2B	60	5F	58	3F	FD	CC	FF	40	EE	B2
2	3A	6E	5A	F1	55	4D	A8	C9	C1	0A	98	15	30	44	A2	C2
3	2C	45	92	6C	F3	39	66	42	F2	35	20	6F	77	BB	59	19
4	1D	FE	37	67	2D	31	F5	69	A7	64	AB	13	54	25	E9	09
5	ED	5C	05	CA	4C	24	87	BF	18	3E	22	F0	51	EC	61	17
6	16	5E	AF	D3	49	A6	36	43	F4	47	91	DF	33	93	21	3B
7	79	B7	97	85	10	B5	BA	3C	B6	70	D0	06	A1	FA	81	82
8	83	7E	7F	80	96	73	BE	56	9B	9E	95	D9	F7	02	B9	A4
9	DE	6A	32	6D	D8	8A	84	72	2A	14	9F	88	F9	DC	89	9A
A	FB	7C	2E	C3	8F	B8	65	48	26	C8	12	4A	CE	E7	D2	62
B	0C	E0	1F	EF	11	75	78	71	A5	8E	76	3D	BD	BC	86	57
C	0B	28	2F	A3	DA	D4	E4	0F	A9	27	53	04	1B	FC	AC	E6
D	7A	07	AE	63	C5	DB	E2	EA	94	8B	C4	D5	9D	F8	90	6B
E	B1	0D	D6	EB	C6	0E	CF	AD	08	4E	D7	E3	5D	50	1E	B3
F	5B	23	38	34	68	46	03	8C	DD	9C	7D	A0	CD	1A	41	1C

AES S-Box

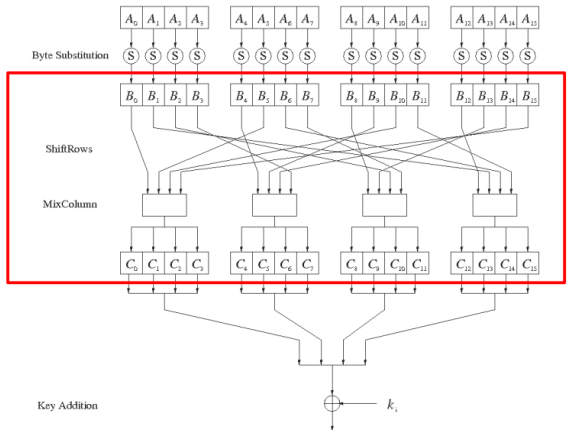
x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Camada de Difusão

A camada de **Difusão**:

- fornece difusão sobre todos os bits do estado de entrada
- consiste em duas subcamadas:
  - **ShiftRows**: Permutação dos dados no nível de bytes
  - **MixColumn**: Operação matricial que combina ("mistura") blocos de quatro bytes
- realiza uma operação linear nas matrizes de estado A, B, ou seja,

$$\text{DIFF}(A) + \text{DIFF}(B) = \text{DIFF}(A + B)$$



Subcamada ShiftRows

- As linhas da matriz de estado são deslocadas ciclicamente:

Matriz de entrada

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix}$$

Matriz de saída

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_5 & B_9 & B_{13} & B_1 \\ B_{10} & B_{14} & B_2 & B_6 \\ B_{15} & B_3 & B_7 & B_{11} \end{bmatrix}$$

- Primeira linha: nenhum deslocamento
- Segunda linha: deslocamento de uma posição à esquerda
- Terceira linha: deslocamento de duas posições à esquerda
- Quarta linha: deslocamento de três posições à esquerda

Subcamada MixColumn

- Transformação linear que mistura cada coluna da matriz de estado
- Cada coluna de 4 bytes é considerada como um vetor e multiplicada por uma matriz fixa 4x4, por exemplo:

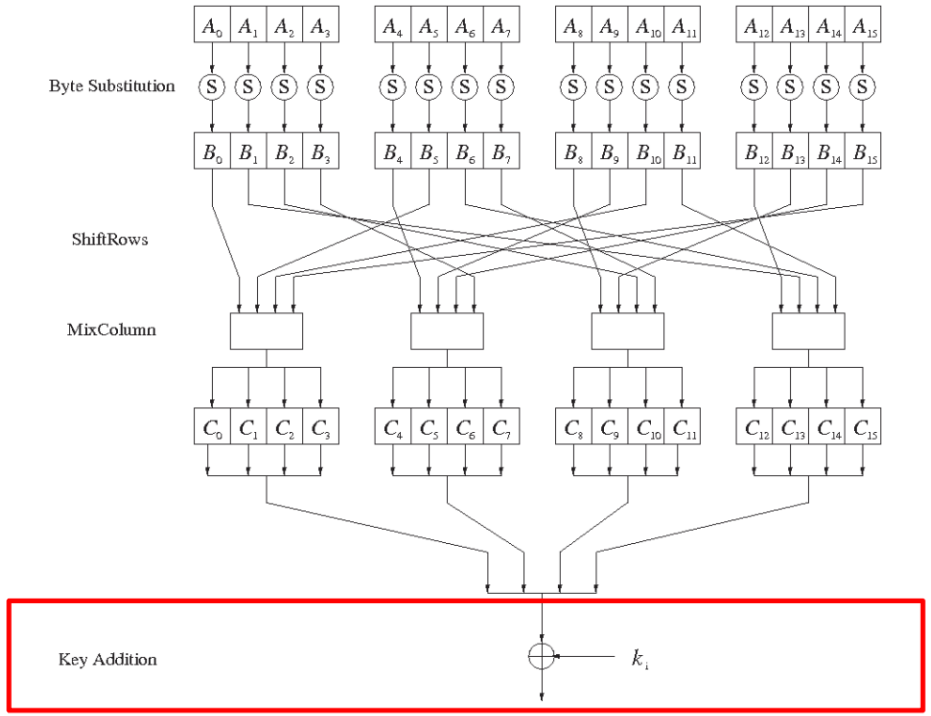
$$\begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{pmatrix}$$

onde 01, 02 e 03 estão em notação hexadecimal.

- Toda a aritmética é feita no campo de Galois $GF(2^8)$

Camada de Adição de Chave

- Entradas:
  - Matriz de estado C de 16 bytes
  - Subchave k_i de 16 bytes
- Saída: $C \oplus k_i$
- As subchaves são geradas no *key schedule*



Key Schedule

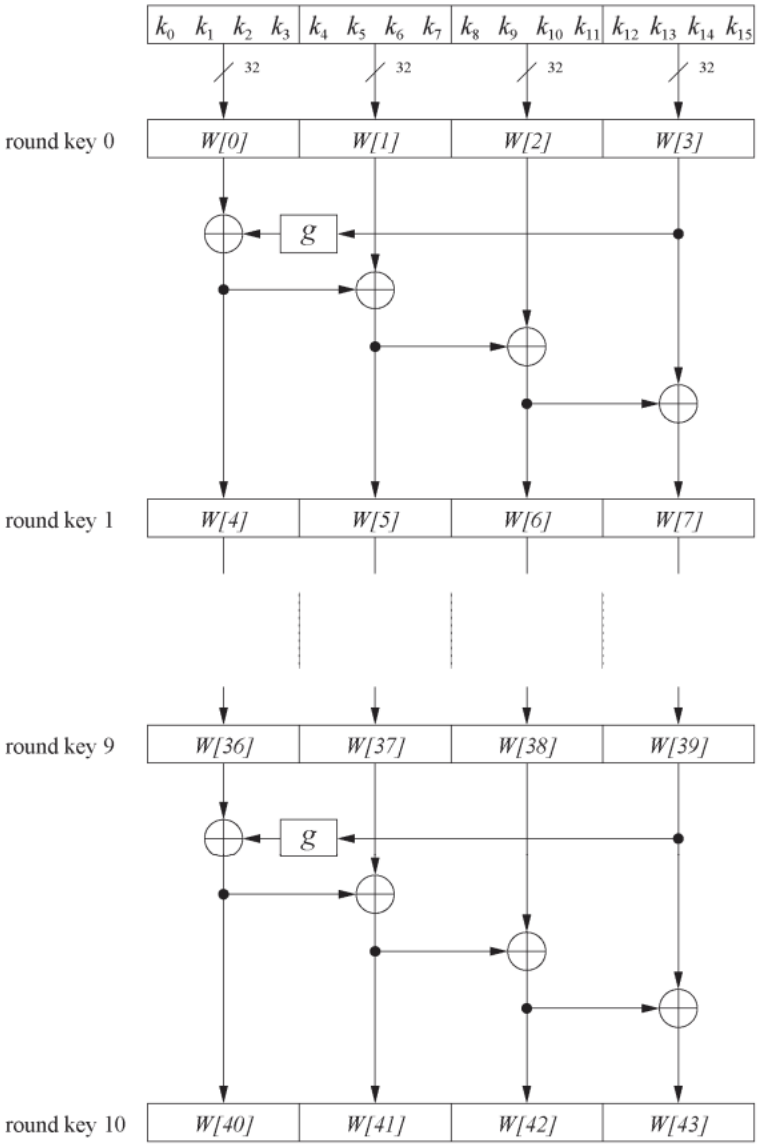
- As subchaves são derivadas recursivamente da chave de entrada original de 128/192/256 bits
- Cada rodada possui 1 subchave, mais 1 subchave no início do AES

Comprimento da chave (bits)	Número de subchaves
128	11
192	13
256	15

- Existem diferentes *key schedules* para os diferentes tamanhos de chave

Key Schedule

Exemplo: *Key schedule* para chave AES de 128 bits



- Orientado a palavras: 1 palavra = 32 bits
- 11 subchaves são armazenadas em $W[0] \ldots W[3]$, $W[4] \ldots W[7]$, ..., $W[40] \ldots W[43]$
- Primeira subchave $W[0] \ldots W[3]$ é a chave AES original

Key Schedule

- A função g rotaciona seus quatro bytes de entrada e realiza uma substituição *bytewise* usando a S-Box $\Rightarrow$ não linearidade.
- O coeficiente de rodada RC é adicionado apenas ao byte mais à esquerda e varia a cada rodada:

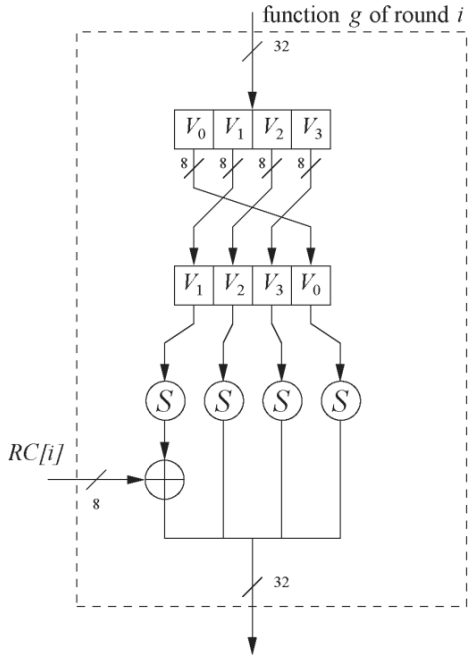
$$RC[1] = x^0 = (00000001)_2$$

$$RC[2] = x^1 = (00000010)_2$$

$$RC[3] = x^2 = (00000100)_2$$

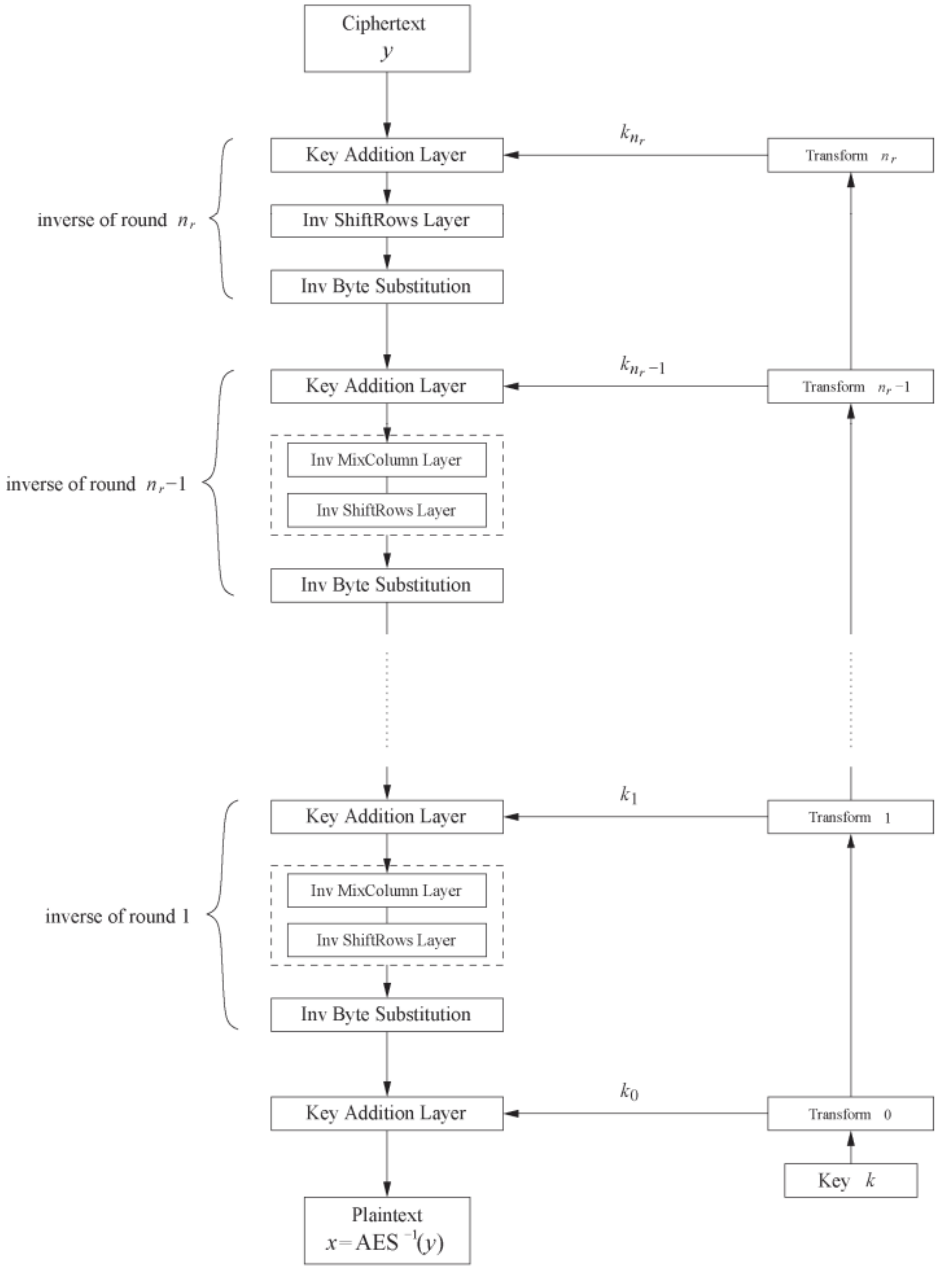
$$\ldots$$

$$RC[10] = x^9 = (00110110)_2$$



- x^i representa um elemento em um campo de Galois

Decryption



- AES não é baseado em uma rede de Feistel $\Rightarrow$ *todas as camadas devem ser invertidas para decriptação:*
- Camada MixColumn $\rightarrow$ **Inv MixColumn layer**
- Camada ShiftRows $\rightarrow$ **Inv ShiftRows layer**
- Camada de SubBytes $\rightarrow$ **Inv SubBytes layer**
- Camada de Adição de Chave é sua própria inversa

Decryption

- **Inv MixColumn layer:**
  - Para reverter a operação MixColumn, cada coluna da matriz de estado C deve ser multiplicada pela **inversa da matriz 4x4**, por exemplo:

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \cdot \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix}$$

onde 09, 0B, 0D e 0E estão em notação hexadecimal.

- Toda a aritmética é realizada no campo de Galois $GF(2^8)$

Decryption

- **Inv ShiftRows layer:**
  - Todas as linhas da matriz de estado B são deslocadas na direção oposta:

Matriz de entrada

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix}$$

Matriz de saída

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_{13} & B_1 & B_5 & B_9 \\ B_{10} & B_{14} & B_2 & B_6 \\ B_7 & B_{11} & B_{15} & B_3 \end{bmatrix}$$

- Primeira linha: nenhum deslocamento
- Segunda linha: deslocamento de uma posição à direita
- Terceira linha: deslocamento de duas posições à direita
- Quarta linha: deslocamento de três posições à direita

Decryption

- **Inv Byte Substitution layer:**
  - Como a S-Box é bijetiva, é possível construir um inverso, tal que:

$$A_i = S^{-1}(B_i) = S^{-1}(S(A_i))$$

$\Rightarrow$ A S-Box inversa é usada para decriptação. Normalmente é implementada como uma *lookup table*.

- **Decryption key schedule:**
  - As subchaves são necessárias em ordem inversa (comparado com a encriptação)
  - Na prática, para encriptação e decriptação, o mesmo *key schedule* é usado.
  - Isso requer que todas as subchaves sejam computadas antes que a encriptação do primeiro bloco possa começar.

Índice

Implementação em Software

- Um dos requisitos do AES foi a possibilidade de uma implementação eficiente em software
- A implementação direta é bem adequada para processadores de 8 bits (por exemplo, *smart cards*), mas ineficiente em processadores de 32 ou 64 bits
- Uma abordagem mais sofisticada: unir todas as funções de rodada (exceto a adição de chave) em uma única *look-up table*
  - Isso resulta em quatro tabelas com 256 entradas, onde cada entrada tem 32 bits
  - Uma rodada pode ser computada com 16 *table look-ups*
- As velocidades típicas de software são superiores a 1,6 Gbit/s em processadores modernos de 64 bits

Segurança

- **Brute-force attack:** Devido ao comprimento da chave de 128, 192 ou 256 bits, um ataque de força bruta não é viável
- **Analytical attacks:** Não existe nenhum ataque analítico conhecido que seja melhor do que a força bruta
- **Side-channel attacks:**
  - Diversos *side-channel attacks* já foram publicados
  - Note que os *side-channel attacks* não atacam o algoritmo em si, mas a sua implementação

Lições Aprendidas

- O AES é uma cifra de bloco moderna que suporta três tamanhos de chave: 128, 192 e 256 bits. Proporciona excelente segurança a longo prazo contra ataques de força bruta.
- O AES tem sido estudado intensivamente desde o final da década de 1990 e nenhum ataque melhor que a força bruta foi encontrado.
- O AES não é baseado em redes de Feistel. Suas operações básicas utilizam aritmética em campos de Galois e proporcionam forte difusão e confusão.
- O AES faz parte de diversos padrões abertos, como IPsec ou TLS, além de ser o algoritmo de criptografia obrigatório para aplicações do governo dos EUA. É provável que a cifra continue sendo o algoritmo de criptografia dominante por muitos anos.
- O AES é eficiente tanto em software quanto em hardware.